Project3——矩阵乘法

5142029014　高超

一、实验内容

给定 M 行、K 列的矩阵 A 和 K 行、N 行的矩阵 B， A 和 B 的矩阵积为矩阵 C，要求是计算矩阵 C。

对于该项目，计算每个 Ci,j 是一个独立的工作线程，因此他会涉及 MxN 个工作线程。主线程将初始化矩阵 A 和 B，并分配足够的内存给矩阵 C, 它会容纳矩阵 A 和 B 的积。

二、实验环境

虚拟机上安装 Ubuntu 15.04 64 位。

三、实验的具体实现

本次实验主要练习多线程，将结果矩阵的每一个数的计算都使用一个线程来实现。使用 pthread_create 函数创建线程。

```
        rc = pthread_create(&thread[i][j], NULL, runner, data);
```

runner 函数进行运算。

然后使用 pthread_join 函数等待工作线程的结束。

```
for(i = 0; i < M; ++i)
    for(j = 0 ; j < N; ++j)
    {
        pthread_join(thread[i][j], NULL);
    }
```

总体上比较简单，于是将 windows 下多线程程序也实现了。与 Linux 下比较类似。

使用 CreateThread 函数创建工作线程。

```
        hthread[i * N + j] = CreateThread(NULL, 0, ThreadProc, data, 0, NULL);
```

Win 32API 提供 WaitForMultipleObjects 函数等待多个工作线程结束。

```
WaitForMultipleObjects(M*N, hthread, TRUE, INFINITE);//等待线程结束
```

四、实验结果的呈现

Linux 下

```
please input the M K N
3 4 4
input the matrix1:
23 45 6 6
35 6  4 7
45 65 34 6

input the matrix
6 57 87 34
7 8 44 92
43 6 7 5
7 8 5 3

the result:
753 1755 4053 4970
473 2123 3372 1783
2229 3337 7043 7698
```

Windows 下

```
input M K N:
3 4 3
input matrix1:
53 65 7 5
75 8 78 5
68 8 87 4

input matrix2:
3 5 7
7 68 9
5 78 8
58 8 3

the result:
939 5271 1027
961 7043 1236
927 7702 1256

--------------------------------
```

五、实验思考与讨论

本次实验比较顺利，理解了多线程以后，懂得如何使用创建线程函数和等待线程函数后，写出程序还是比较容易的。

但是，事实上，本来希望能够体现出多线程的优越性，于是想写一个不用多线程的程序来比较程序运行时间，但是实际上差距并不大，思考原因，可能是创建线程，取消线程也需要时间导致差异不明显，同时由于使用虚拟机（分配了双核），即使数字取得比较大，差距不明显，而且感觉 cpu 在执行过一次以后，再次执行程序速度明显快了，于是最终没有很好的解释。听到有同学建立数学模型来进行多次尝试后的数据分析，不得不心生敬佩，所以以后还有更多需要学习的地方。